View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Repositório Científico do Instituto Politécnico do

Combined Mutation Differential Evolution to Solve Systems of Nonlinear Equations

Gisela C.V. Ramadas* and Edite M.G.P. Fernandes†

*Department of Mathematics, School of Engineering, Polytechnic of Porto, 4200-072 Porto, Portugal
†Algorithm R&D Center, University of Minho, 4710-057 Braga, Portugal

Abstract. This paper presents a differential evolution heuristic to compute a solution of a system of nonlinear equations through the global optimization of an appropriate merit function. Three different mutation strategies are combined to generate mutant points. Preliminary numerical results show the effectiveness of the presented heuristic.

Keywords: Nonlinear Equations, Differential Evolution, Combined Mutation
PACS: 02.60.Pn

INTRODUCTION

The primary goal of the paper is to show that an evolutionary heuristic – the differential evolution – very popular in global optimization can be effective and as efficient as classical methods in solving systems of nonlinear equations. We examine the behavior of different mutation strategies in the differential evolution context to solve a system of the form:

$$f(x) = 0,\ f(x) = (f_1(x), f_2(x), \ldots, f_n(x))^T \tag{1}$$

where each $f_i : \Omega \subset \mathbb{R}^n \to \mathbb{R}$ and Ω is a closed convex set, herein defined as $[l,u] = \{x : -\infty < l_i \leq x_i \leq u_i < \infty, i = 1,\ldots,n\}$. We assume that all functions $f_i(x)$, $i = 1,\ldots,n$ are continuous in the search space although differentiability may not be guaranteed. The motivation of this work comes mainly from the detection of feasibility in nonlinear optimization problems. The most famous techniques to solve nonlinear equations are based on the Newton's method [1, 2]. They require analytical or numerical first derivative information. Newton's method is the most widely used algorithm for solving nonlinear systems of equations. It is computationally expensive, in particular if n is large, since the Jacobian matrix and the solution of a system of linear equations are required at each iteration. On the other hand, Quasi-Newton methods avoid either the necessity of computing derivatives, or the necessity of solving a full linear system per iteration or both tasks [3]. Thus, Quasi-Newton methods have less expensive iterations than Newton, and their convergence properties are not very different from the Newton one.

Problem (1) is equivalent to

$$\min_{x \in \Omega \subset \mathbb{R}^n} \mathscr{M}(x) \equiv \sum_{i=1}^{n} f_i(x)^2, \tag{2}$$

in the sense that they have the same solutions. These required solutions are the global minima, and not just the local minima, of the function $\mathscr{M}(x)$, known as merit function, in the set Ω. Problem (2) is similar to the usual least squares problem for which many iterative methods have been proposed. They basically assume that the objective function is twice continuously differentiable. However, the objective $\mathscr{M}$ in (2) is only once differentiable if some, or just one, of the f_i, $(i = 1,\ldots,n)$ are not differentiable. Thus, methods for solving the least squares problem cannot be directly applied to solve (2). When a global solution of a nonlinear optimization problem is required, Newton-type methods have some disadvantages, when compared with global search methods, because they rely on searching locally for the solution. The final solution is heavily dependent on the initial approximation of the iterative process and they can be trapped in a local minimum.

Preventing premature convergence to a local while trying to locate a global solution of problem (2) is the goal of the present study. Here, we aim to investigate the performance of a new version of the differential evolution (DE) algorithm when globally solving problem (2). DE is a population-based evolutionary algorithm introduced in 1997 by Storn and Price [4]. It is a simple, efficient and robust metaheuristic to search for promising regions and locate a global solution. Our proposal joins three mutation strategies. It combines two classic mutation strategies aiming to explore

11th International Conference of Numerical Analysis and Applied Mathematics 2013
AIP Conf. Proc. 1558, 582-585 (2013); doi: 10.1063/1.4825558
© 2013 AIP Publishing LLC 978-0-7354-1184-5/$30.00

the search space and makes use cyclically of another strategy to exploit a promising region and accelerate convergence. Although other metaheuristics have been proposed to solve systems of nonlinear equations, the computational effort to achieve a solution is huge. Further, the quality of the solution is not in general satisfactory. A genetic algorithm is proposed in [5]. Other stochastic approaches have been used to compute a root of the system (1) by solving the global optimization problem (2). A tabu search based framework has been implemented together with a local search strategy to enhance the search about a promising region and improve the quality of the solution [6, 7]. In [8], a new technique for solving systems of nonlinear equations reshaping the system as a multiobjective optimization problem is proposed. A technique of evolutionary computation is then applied to solve the multiobjective problem.

The remaining paper is organized as follows. Firstly, the main steps of the basic DE algorithm as well as the ideas of the proposed combined mutation strategy are described. Afterwards, a discussion about the obtained numerical results is included.

DIFFERENTIAL EVOLUTION

DE is a simple evolutionary algorithm for global optimization problems introduced by Storn and Price [4]. The initial population is randomly generated in the search space Ω. After initialization DE employs three operations – *mutation*, *crossover* and *selection* – until a global solution is reached. Based on the current population of m points, x^j, $j = 1,\ldots,m$, called target points, DE generates new solutions by combining the current solutions and several other solutions of the same population. DE has three parameters: i) amplification factor of the differential variation, F; ii) crossover control parameter, CR; iii) population size, m. The above three operations are repeated until a termination criterion is reached. A survey with an experimental study concerned with variants of DE is found in [9]. In this section, we describe the main steps of the basic DE and a modified operation of mutation that combines three classic mutation strategies. The below presentation already takes into consideration the global optimization problem that we aim to address (2).

Basic DE

Algorithm 1 contains the pseudocode of the basic DE algorithm. The most commonly used *mutation* is referred to

Algorithm 1 DE algorithm

Step 0.	Set values to the parameters. Set $k = 0$.
Step 1.	Randomly generate the (population) target points $x^i \in \Omega$, $i = 1\ldots,m$.
Step 2.	Evaluate the population and select x^{best}.
Step 3.	Perform *mutation* to generate the mutant points.
Step 4.	Perform *crossover* to generate the trial points. Check with the bounds and project if necessary. Evaluate the trial points.
Step 5.	Perform *selection* to define the target points for the next population and select x^{best}.
Step 6.	If x^{best} is sufficiently accurate then STOP else increase k and go to Step 3.

as DE/rand/1 and defines the mutant point, v^j, as follows:

$$v^j = x^{r_1} + F(x^{r_2} - x^{r_3}) \tag{3}$$

with uniformly chosen random indices r_1, r_2, r_3 from the set $\{1,2,\ldots,m\}$, mutually different and F is a real parameter $\in [0,2]$ which controls the amplification of the differential variation, $x^{r_2} - x^{r_3}$. The indices r_1, r_2 and r_3 are also chosen to be different from the index j. x^{r_1} is called the base point. This version is denoted hereafter by DE$_{rand}$. There are other frequently used mutation strategies, for instance, the DE/best/1, herein denoted by DE$_{best}$, which uses the best point of the population as the base point:

$$v^j = x^{best} + F(x^{r_1} - x^{r_2}) \tag{4}$$

where x^{best} is the best point in the current population. The *crossover* operator aims to increase the diversity on the components of the mutant point. Thus, the crossover point, called trial point, y^j, is formed as:

$$y_i^j = \begin{cases} v_i^j & \text{if } rand() \le CR \text{ or } i = s_j \\ x_i^j & \text{otherwise} \end{cases} \tag{5}$$

for $i = 1,\ldots,n$, where $rand()$ denotes a random number in $[0,1]$ and aims to perform the mixing of the component i of the points, $CR \in [0,1]$ is the parameter for crossover, and the index s_j, randomly selected from $\{1,\ldots,n\}$, ensures that y^j gets at least one component from v^j. When generating the mutant point, some components can be generated outside the domain Ω. Thus, each component should be checked and a projection to the bounds is to be carried out:

$$y_i^j = \max\left\{l_i, \min\left\{y_i^j, u_i\right\}\right\}, \text{ for } i = 1,\ldots,n.$$

To perform *selection*, the trial point is then compared with the target and if the merit function value of the trial point is better than that of the target, the trial will be the target for the next population. DE performance depends on the amplification factor of differential variation, F, and crossover control parameter, CR. It is not an easy task to set appropriate values for the parameters F and CR, since they depend on the nature and size of the optimization problem, A self-adaptive technique to control the parameters and generate a different set of F^j, CR^j for each point has been implemented in DE [10]. Other frequently used mutation strategies can be found in [11].

Combined Mutation DE

The performance of DE depends mostly on its ability to explore the entire search space as well as to exploit around the neighborhood of a reference point, that could be the best point of the population. The DE/rand/1 mutation strategy has an exploratory effect but it slows down the convergence of DE. On the other hand, DE/best/1 is good at accelerating convergence but may be poor at exploring the space and a local solution may be obtained before the global solution can be reached. Thus, a proper balance between exploration and exploitation is required for an effective mutation operation. A modified mutation that mixes three mutation strategies is presented. The first mutation strategy is a combination of two different strategies with a weight factor and the second is concerned with a cyclical use of DE/best/1 strategy. This combined mutation DE version will be called *comb*-DE. Firstly, three points are randomly chosen from the target population and the first mutant point, $\bar{v}^j$, is created using (3). Secondly, the DE/current-to-best/1 mutation strategy is used to create the second mutant point:

$$\tilde{v}^j = x^j + F(x^{best} - x^j) + F(x^{r_4} - x^{r_5}) \tag{6}$$

where uniformly chosen random indices r_4, r_5 from the set $\{1,\ldots,m\}$ are mutually different and also chosen to be different from r_1, r_2, r_3 (see (3)) and the index j. Finally, the actual mutant point v^j is obtained by combining the above two mutant points using a scalar weight factor $w \in [0,1]$

$$v^j = w\bar{v}^j + (1-w)\tilde{v}^j. \tag{7}$$

We note that m must be greater or equal to six to allow for this condition. Furthermore, at every R iterations, we use (4) to define the current mutant point.

NUMERICAL RESULTS AND DISCUSSION

In this comparative study, four well-known small-dimensional problems and two large-dimensional problems, using different values of n, are used [12]. The results of these experiments were obtained in a personal computer with an AMD Turion 2.20 GHz processor and 3 GB of memory, and all program codes were written in MATLAB R2010b. During the experiments, we set $m = \min\{3n, 20\}$, $F = 0.9$, $CR = 0.9$, and tested the sets $w = \{0.15, 0.5, 0.85\}$ and $R = \{10, 50\}$.

Table 1 contains a summary of the results, in terms of number of function evaluations of the best run, 'nfe_{best}', among 10 independent runs for each problem. The algorithms in comparison were terminated when $\|f(x^{best})\|_2 \leq 10^{-6}$ (or, $\mathscr{M}(x^{best}) \leq 10^{-12}$ equivalently) is satisfied for the best of the m points, or the number of function evaluations reached the target value '$nfe_{\max}$'. The algorithm was allowed to run for a maximum of 10 000 function evaluations, when solving the problems `Reactor` and `Steering`, a maximum of 1 000 function evaluations when solving `Merlet` and `Floudas` and 100 000 function evaluations, when solving `Yamamutra` (n=10, 20, 30, 40) and `Bratu` (n=10, 20, 30). We observe that apart from the DE$_{best}$, the pairs $(0.15, 10)$, $(0.85, 10)$, $(0.15, 50)$ and $(0.85, 50)$ for (w, R) in the *comb*-DE context are the most effective strategies for solving the reported test problems, and the versions *comb*-DE with $w = 0.15$ require in general less function evaluations than the other versions in comparison.

These preliminary results are very encouraging and future research will be focused on the sensitivity analysis of the *comb*-DE algorithm to the values of some parameters, like F and CR. Furthermore, this proposed *comb*-DE algorithm will be applied to solve systems of inequalities and equalities, as well as nonlinear complementarity problems.

TABLE 1. Results of nfe_{best}.

Prob.	n	Ω	DE_{rand}	DE_{best}	*comb*-DE as a function of (w,R)				
					$(0.15,10)$	$(0.5,10)$	$(0.85,10)$	$(0.15,50)$	$(0.85,50)$
Reactor	2	$[0,1]^2$	678	606	438	(3e-3)	(3e-3)	(3e-3)	(3e-3)
Steering	3	$[0.06,1]^3$	936	486	(6e-8)	(6e-8)	(6e-8)	(6e-8)	(6e-8)
Merlet	2	$[0,2\pi]^2$	12	12	12	18	18	24	24
Floudas	2	$[0.25,1]\times[1.5,2\pi]$	642	444	324	258	450	282	576
Yamamutra	10	$[0,1]^{10}$	19320	5920	2640	2900	8340	3900	8820
	20	$[0,1]^{20}$	48240	17380	9700	(3e-7)	18260	8760	17900
	30	$[0,1]^{30}$	85720	31920	15560	(1e-5)	29400	16420	29600
	40	$[0,1]^{40}$	(2e-11)	43600	27920	(1e-3)	44520	29660	41740
Bratu	10	$[0,1]^{10}$	45000	24220	14980	(1e-9)	18660	19460	23360
	20	$[0,1]^{20}$	(8e-5)	(1e-4)	(8e-5)	(5e-3)	(3e-5)	(6e-5)	(3e-6)
	30	$[0,1]^{30}$	(3e-5)	(3e-5)	(5e-4)	(1e-2)	(7e-5)	(6e-4)	(4e-4)

NOTE: values in parentheses correspond to $\mathscr{M}(x^{best})$ after $nfe_{\max}$ function evaluations.

ACKNOWLEDGMENTS

This research has been supported by CIDEM (Centre for Research & Development in Mechanical Engineering, Portugal), FCT (Foundation for Science and Technology, Portugal) and FEDER COMPETE (Operational Programme Factors of Competitiveness) under projects PEst-OE/EME/UI0615/2011 and FCOMP-01-0124-FEDER-022674.

REFERENCES

1. J.E. Dennis, R.B. Schnabel, *Numerical Methods for Unconstrained Optimization and Nonlinear Equations*, Prentice-Hall Inc., 1983.
2. A. Friedlander, M.A. Gomes-Ruggiero, D.N. Kozakevich, J.M. Martínez, S.A. Santos, Solving nonlinear systems of equations by means of Quasi-Newton methods with a nonmonotone strategy, *Optimization Methods and Software* 8, 25–51 (1997)
3. J.M. Martínez, Practical Quasi-Newton methods for solving nonlinear systems, *Journal of Computational and Applied Mathematics* 124, 97–122 (2000).
4. R. Storn, K. Price, Differential evolution – a simple and efficient heuristic for global optimization over continuous spaces, *Journal of Global Optimization* 11, 341–359 (1997).
5. C.H. Chen, Finding Roots By Genetic Algorithms, in *2003 Joint Conference on AI, Fuzzy System and Gray System*, Taipei, Taiwan, 4–6 (2003).
6. G.C.V. Ramadas, E.M.G.P. Fernandes, Self-adaptive combination of global tabu search and local search for nonlinear equations, *International Journal of Computer Mathematics* 89(13–14), 1847–1864 (2012).
7. G.C.V. Ramadas, E.M.G.P. Fernandes, Nonmonotone hybrid tabu search for inequalities and equalities: an experimental study, *Applied Mathematical Sciences* 7(11), 503–525 (2013).
8. C. Grosan, A. Abraham, A new aproach for solving nonlinear equations systems, *IEEE Transactions on Systems, Man and Cybernetics - Part A: Systems and Humans* 38(3), 698–714 (2008).
9. F. Neri, V. Tirronen, Recent advances in differential evolution: a survey and experimental analysis, *Artificial Intelligence Review* 33(1-2), 61–106 (2010).
10. J. Brest, S. Greiner, B. Bošković, M. Mernik, V. Žumer, Self-adapting control parameters in differential evolution: a comparative study on numerical benchmark problems, *IEEE Transaction on Evolutionary Computation* 10, 646–657 (2006).
11. R. Mallipeddi, P.N. Suganthan, Q.K. Pan, M.F. Tasgetiren, Differential evolution algorithm with ensemble of parameters and mutation strategies, *Applied Soft Computing* 11, 1679–1696 (2011).
12. I.G. Tsoulos, A. Stavrakoudis, On locating all roots of systems of nonlinear equations inside bounded domain using global optimization methods, *Nonlinear Analysis: Real World Applications* 11, 2465–2471 (2010).